Cadenas de texto
En este módulo, se aprenderá a crear controles e indicadores para cadenas de texto.

Una cadena de texto es una secuencia de caracteres ASCII ya sean visualizables o no. Las cadenas proporcionan una forma de mostrar información independientemente de la plataforma utilizada. Los usos más comunes de las cadenas de texto son:

- Crear simples mensajes de texto.
- Entregar datos numéricos a instrumentos como cadenas de caracteres para después convertir las cadenas en valores numéricos.
- Almacenar datos numéricos en disco. Para almacenar valores numéricos en un fichero ASCII, hay que convertir los valores numéricos en cadenas de texto antes de escribirlos en el fichero.
- Dar avisos o pedir datos al usuario mediante cajas de dialogo.

En el panel frontal, las cadenas de texto aparecen como tablas, cajas de entrada de texto y etiquetas.

Crear String Controls e Indicators

Utilizar los String Controls e Indicators situados en las paletas `Controls>>Text Controls` y `Controls>>Text Indicators` para simular cajas de entrada de texto y etiquetas. Usar la herramienta de operaciones o etiquetado para escribir o editar texto en un String Control. Usar la herramienta de posicionado para redimensionar el objeto cadena del panel frontal. Para minimizar el espacio que ocupa un objeto cadena, hacer clic con el botón derecho sobre el objeto y seleccionar en el menú desplegable la opción `Visible Items>>Scrollbar`.

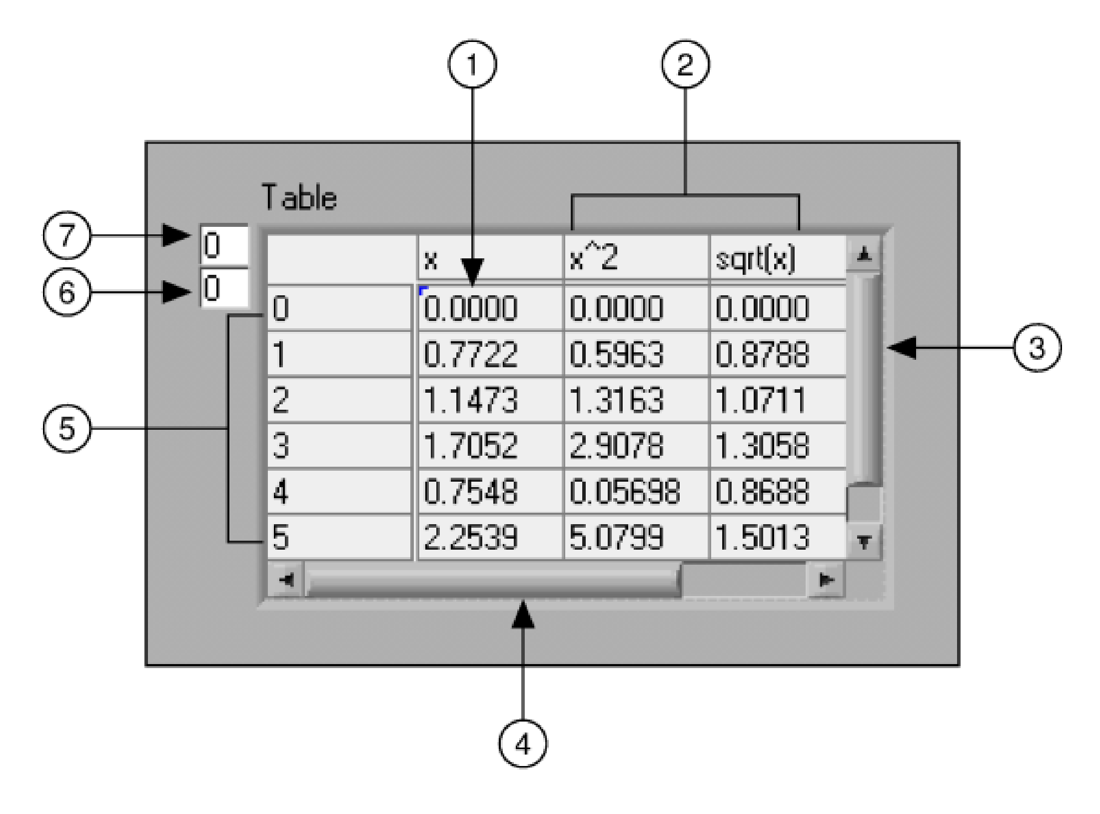
Hacer clic con el botón derecho sobre un String Control o Indicator en el panel frontal para seleccionar de entre los tipos de visualización mostrados en la [link]. La tabla también muestra un mensaje ejemplo para cada tipo de visualización.

Display Type	Description	Message
Normal Display	Muestra los caracteres imprimibles usando la fuente del control. Los caracteres no imprimibles aparecen generalmente como cajitas. Hay cuatro tipos de visualización.	`There are four display types.` `\ is a` `backslash.`
'\' Codes Display	Muestra códigos de backslash para todos los caracteres no visualizables.	`There\sare\sfour\sdisplay\s` `types.\n\\\sis\sa\sbackslash.`
Password Display	Muestra un asterisco (*) por cada carácter incluidos los espacios.	`***************************` `*******************`

Display Type	Description	Message
Hex Display	Muestra los valores ASCII de cada carácter en hexadecimal en vez del caracter propiamente.	`5468 6572 6520 6172 6520 666F 7572 2064 6973 706C 6179 2074 7970 6573 2E0A 5C20 6973 2061 2062 6163 6B73 6C61 7368 2E`

Tablas

Usar el control Table situado en la paleta `Controls>>All Controls>>List & Table` o el instrumento virtual `Express Table` situado en la paleta `Controls>>Text Indicators` para crear una tabla en el panel frontal. Cada celdilla en la tabla es una cadena, y cada cadena reside en una fila y una columna. Por lo tanto, una tabla es un visualizador para un array de cadenas de texto de 2 dimensiones. La ilustración mostrada en la [link] muestra una tabla y todos sus elementos.



1. Celdillas apuntadas por los Índices, 2. Encabezamiento de las Columnas, 3. Barra de desplazamiento Vertical, 4. Barra de desplazamiento Horizontal, 5. Encabezamiento de las Filas, 6. Índice Horizontal, 7. Índice Vertical.

Definir las celdillas de una tabla usando la herramienta `Operating` o la herramienta `Labeling` para seleccionar una celdilla y escribir el texto en su interior.

La tabla visualiza un array de 2 dimensiones de cadenas de texto, así que para poder visualizar en la tabla un array numérico habrá que convertirlo en un array de caracteres de texto. Los indicadores de fila y columna no se muestran automáticamente como en una hoja de calculo. Hay que crear un array unidimensional de cadenas de texto para los encabezamientos de las filas y las columnas.

Funciones para trabajar con Cadenas de Texto
En este módulo, se aprenderá a hacer uso de las funciones para trabajar con cadenas de texto.

Las funciones de String situadas en la paleta `Functions>>All Functions>>String` se usan para editar y manipular cadenas en el diagrama de bloques. Las funciones de String disponibles son las siguientes:

- **String Length** Devuelve el número de caracteres (bytes) de la **cadena**, incluyendo los caracteres espacio. Por ejemplo, la función `String Length` devuelve una **longitud** de **19** para la siguiente cadena: `The quick brown fox`
- **Concatenate Strings** Encadena cadenas de texto de entrada y arrays unidimensionales de cadenas de texto en una única cadena de salida. Para las entradas del array, esta función encadena cada elemento del array. Añadir entradas a la función haciendo clic con el botón derecho sobre la función y seleccionando Add Input en el menú contextual o también redimensionando la función. Por ejemplo, encadenar la cadena anterior con el siguiente array de cadenas de texto:

`jumped`	`over`	`the`	`lazy`	`dog.`

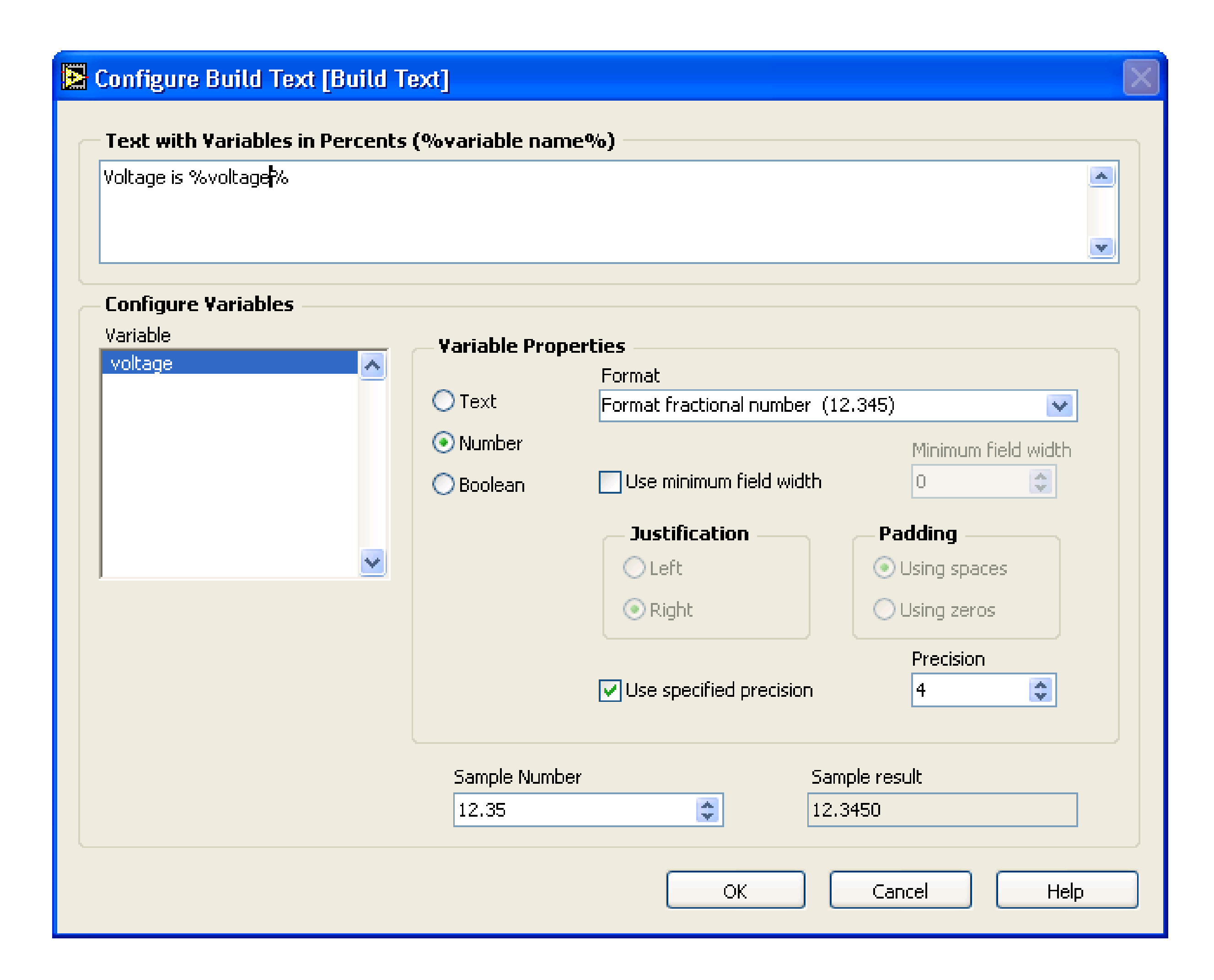
  La función `Concatenate Strings` devuelve la siguiente cadena: `The quick brown fox jumped over the lazy dog.`
- **String Subset** Devuelve la **subcadena** de la **cadena de texto** de entrada comenzando por el carácter de **offset** y conteniendo el número de caracteres indicado en **length**. El primer caracter de la **cadena** tiene un **offset** de `0`. Por ejemplo, si se usa la cadena anterior como entrada, la función `String Subset` devuelve la siguiente **subcadena** para un **offset** de `4` y una **length** de `5`: `quick`

- **Match Pattern** Busca una **expresión regular** en la **cadena de texto** comenzando a partir de un **offset**, y si encuentra una coincidencia, parte la **cadena** en tres subcadenas. Si no se encuentra una coincidencia, **match substring** queda vacío y el **offset past match** a −1. Por ejemplo, usar una **regular expression** de `:` y la siguiente cadena como entrada: `VOLTS DC: +1.22863E+1;` La función `Match Pattern` devuelve **before substring** con `VOLTS DC`, **match substring** con `:`, y **after substring** con `+1.22863E+1;`, y el **offset past match** quedará con `9`.

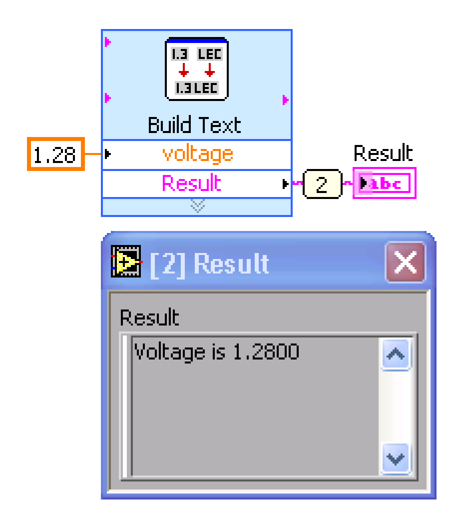
Conversión de valores numéricos a cadenas de texto con el instrumento virtual Express Build VI

Usar el `Build Text` Express VI para convertir valores numéricos en cadenas de texto. El `Build Text` Express VI, situado en la paleta `Functions>>Output`, encadena una cadena de entrada. Si la entrada no es una cadena de texto, este Express VI convierte la entrada en una cadena, basándose en la configuración del Express VI.

Cuando se coloca un `Build Text` Express VI en el diagrama de bloques, aparece la caja de dialogo Configure Build Text. Esta caja de dialogo muestra el VI Express configurado para aceptar una entrada, `voltage`, y cambiarla a un número fraccional con una precisión de 4 dígitos. La entrada se encadena al final de la cadena `Voltage is`. Se ha incluido un espacio al final de la cadena `Voltage is`.



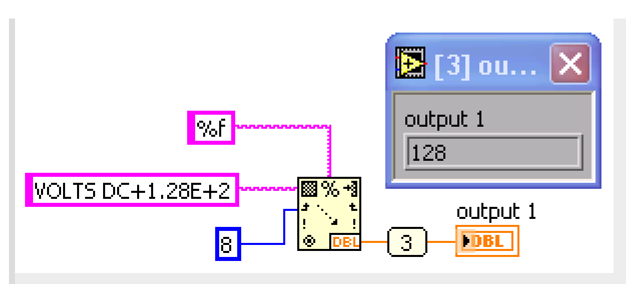
Esta configuración produce el diagrama de bloques mostrado en la [link]. Se ha añadido la ventana que muestra el valor de la cadena de salida. El VI Express `Build Text` encadena la entrada `Beginning Text`, en este caso el valor del `voltaje`, al final del texto configurado.



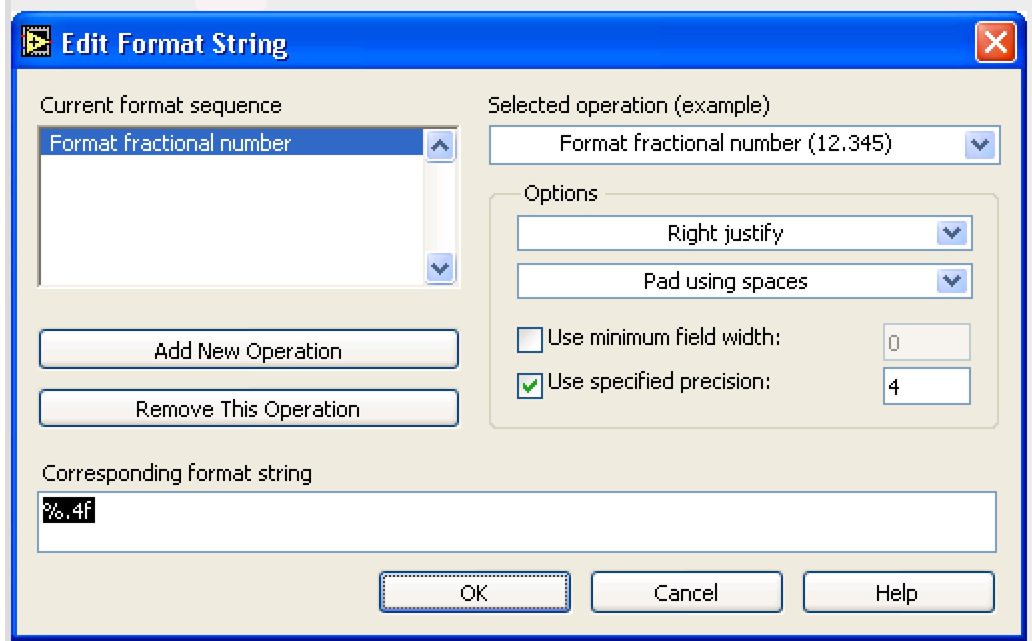
Conversión de cadenas de texto a valores numéricos con la función Scan From String

La función `Scan From String` convierte una cadena conteniendo caracteres numéricos válidos, tales como `0-9`, `+`, `-`, `e`, `E`, y punto decimal (`.`) a un valor numérico. Esta función explora la **cadena de entrada** y la convierte de acuerdo al **formato de la cadena**. Usar esta función cuando se conoce el formato exacto del texto de entrada. Esta función puede escanear la **cadena de entrada** en varios tipos de datos, como Numericos o Booleanos, basándose en el formato de la cadena. Redimensionar la función para incrementar el número de **salidas**.

Example:
Por ejemplo, usar como **formato de cadena %f**, una **posición inicial de búsqueda** de `8`, y `VOLTS DC+1.28E+2` como **cadena de entrada** para obtener una salida de `128`, según se muestra en el diagrama de bloques mostrado en la [link]. Cambiar la precisión de la salida cambiando la precisión del indicador.



En la **cadena de formato**, `%` comienza el especificador de formato y `f` indica un número en coma flotante con formato fraccional. Hacer clic con el botón derecho sobre la función y seleccionar `Edit Scan String` en el menú contextual para crear o editar un formato para la cadena. La caja de diálogo de Edit Scan String muestra la configuración para una cadena de formato `%4f`.



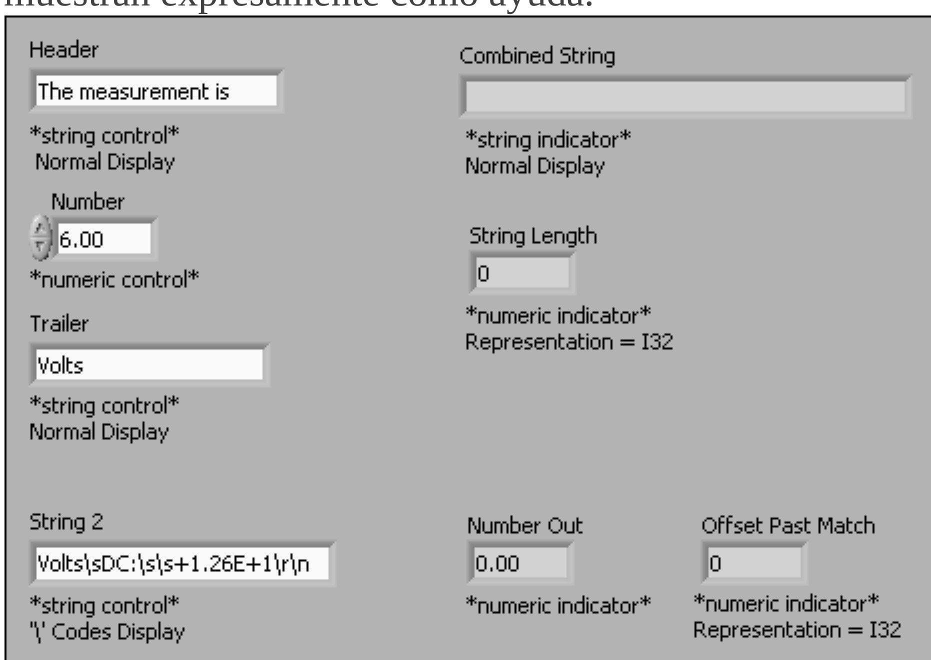
Consultar la ayuda de LabVIEW para una mayor información acerca de la sintaxis de los especificadores de formato.

VI para crear una cadena de texto
En este ejercicio se pretende usar las funciones Build String, Match Pattern, Scan from String, y String Length. Realizar los pasos indicados para construir un VI que convierta valores numéricos a cadenas de texto, que agrupe una cadena de texto junto con otras cadenas en una única cadena de texto, y que determine la longitud de una cadena de salida. Este instrumento virtual también busca un patrón en una cadena de texto y convierte los caracteres restantes en un valor numérico.
Exercise:

Problem:

Panel Frontal

1. Abrir un diseño en blanco y construir el panel frontal mostrado en la [link]. No añadir etiquetas para los comentarios; estos se muestran expresamente como ayuda.



Tener en cuenta las siguientes consideraciones para construir el panel frontal:

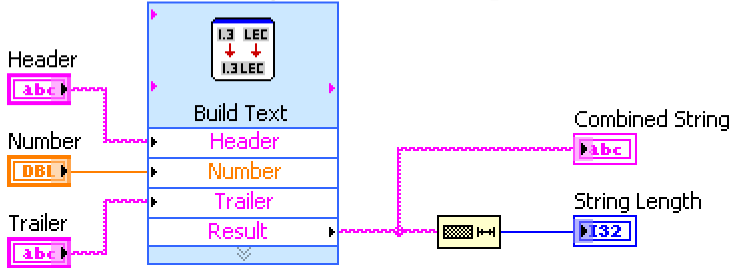
   - Hacer clic con el botón derecho sobre `String 2` y seleccionar '`\`' `Codes Display` en el menú desplegable. El texto introducido en `String 2` es `Volts DC:`

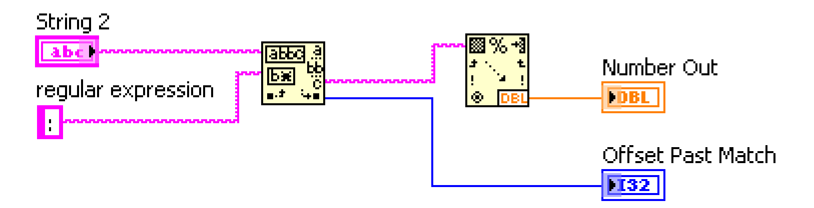
`+1.26E+1`. Como se muestra usando Codes Display en el panel frontal mostrado arriba, hay 2 espacios tras el caracter dos puntos (`\s\s`), y el texto finaliza con un retorno de carro (`\r\n`). También se podría ver un `\r` o un `\n` para el retorno de carro.

- Cambiar la representación de los indicadores `String Length` y `Offset Past Match` a entero con signo de 32 bits (I32).
- Después de introducir el texto en los controles, seleccionar `Operate>>Make Current Values Default` para fijar ese texto como los valores por defecto de esos controles.

Diagrama de Bloques

1. Construir el Diagrama de Bloques mostrado en la [link].





    1.

    Colocar en el diagrama de bloques el instrumento virtual `Build Text Express VI`, situado en la paleta `Functions>>Output`. Esta función convierte `Números`

a una cadena de texto. Al colocar el VI express `Build Text` se muestra la caja de dialogo de configuración.

1. Escribir `%Header% %Number% %Trailer%` en la caja de texto `Text with Variables in Percents` para crear tres variables. Las variables aparecen en la sección `Configure Variables`.
2. Seleccionar `Number` en la sección `Variable`.
3. En la sección `Variable Properties`, seleccionar la opcion `Number`, establecer el formato a `Format fractional number`. Marcar la casilla `Use specified precision` y fijar la `precisión` en `4`. Dejar las variables `Header` y `Trailer` con los valores por defecto.
4. Hacer clic en el botón `OK` para cerrar la caja de dialogo.

2.

Colocar en el diagrama de bloques la función `String Length`, situada en la paleta `Functions>>All Functions>>String`. Esta función devuelve el número de caracteres en `Result`.

3.

Colocar en el diagrama de bloques la función `Match Pattern`, situada en la paleta `Functions>>All Functions>>String`. Esta función busca el carácter dos puntos (`:`) en `String 2`. Hacer clic con el botón derecho sobre la entrada `regular expression`, y seleccionar en el menú que aparece `Create>>Constant`, escribir el carácter dos puntos (`:`), y pulsar la tecla <Enter> en el teclado numérico. También se puede hacer clic en el botón `Enter` de la barra de herramientas para completar la entrada. No pulsar la tecla <Enter> en el teclado principal,

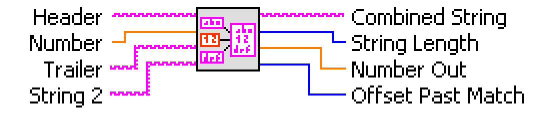
porque en ese caso se introduce el carácter return en la cadena de búsqueda.

4. 

   Colocar en el diagrama de bloques la función `Scan From String`, situada en la paleta `Functions>>All Functions>>String`. Esta función convierte la cadena después del carácter dos puntos en un valor numérico.
5. Completar el diagrama de bloques como se muestra en la [link].

Icono del Instrumento y Panel conector

1. Visualizar el panel frontal y crear un icono y un panel conector, de tal modo que más tarde se pueda usar a lo largo del curso este Instrumento Virtual VI como un subVI. Tener como referencia la lección sobre programacion modular para obtener más información sobre cómo crear iconos y paneles conector.



2. Salvar el Instrumento Virtual VI como `Create String.vi` en el directorio `C:\Exercises\LabVIEW Basics I`. Este VI se usará más adelante en este curso.

Ejecución del VI

1. Cambiar los valores en los controles del panel frontal y ejecutar el VI. El VI concatena `Header`, `Number`, y `Trailer` en una `cadena combinada` y muestra la longitud de la cadena. El VI también busca el carácter dos puntos (`:`) en `String 2`, convierte los caracteres de la cadena que siguen al carácter dos puntos (`:`), los saca en `Number Out`, y muestra el índice al

primer caracter después del carácter dos puntos (`:`) en `Offset Past Match`.

2. Salvar y cerrar el VI.

VIs y funciones para trabajar con ficheros
En este módulo, se aprenderá a realizar operaciones de lectura desde y escritura en ficheros.

Las operaciones de entrada/salida con ficheros transfieren datos desde y hacia los ficheros. Se van a usar los VIs y las funciones disponibles en la paleta `Functions>>All Functions>>File I/O` para gestionar todos los aspectos de entrada y salida con ficheros, incluyendo los siguientes:

- Apertura y cierre de ficheros de datos
- Leer datos desde y escribir datos en ficheros
- Leer desde y escribir en ficheros con formato de hoja de cálculo
- Mover y renombrar ficheros y directorios
- Cambiar las características de los ficheros
- Crear, modificar y leer ficheros de configuración

VIs para File I/O

La paleta File I/O está dividida en 4 tipos de operaciones: alto nivel, bajo nivel, avanzado y express.

VIs de alto nivel para File I/O

Usar los VIs File I/O de alto nivel situados en la fila superior de la paleta `Functions>>All Functions>>File I/O` para ejecutar las tareas habituales de I/O. Consulta la sección High-Level File I/O VIs para obtener más información sobre los instrumentos virtuales para entrada y salida con ficheros.

Se puede ahorrar tiempo y esfuerzo de programación usando los VIs de alto nivel para escribir y leer desde ficheros. Los VIs de alto nivel ejecutan operaciones de lectura y escritura, además de abrir y cerrar el fichero. Si ocurriera un error, los Vis de alto nivel muestran una caja de dialogo que describe el error. Se puede elegir entre detener la ejecución o seguir. Sin embargo, dado que los VIs de alto nivel, encapsulan por completo la

operación con el fichero en un VI, son difíciles de adaptar a cualquier otro uso diferente al indicado. Usar VIs de bajo nivel para tareas más específicas.

VIs y funciones de bajo nivel y avanzadas para File I/O

Usar los VIs y las funciones File I/O de bajo nivel situados en la fila intermedia de la paleta `Functions>>All Functions>>File I/O` y las funciones Advanced File I/O situadas en la paleta `Functions>>All Functions>>File I/O>>Advanced File Functions` para controlar cada una de las operaciones de entrada/salida de ficheros de manera individual.

Usar las funciones más importantes de bajo nivel para crear o abrir un fichero, escribir datos en o leer datos desde el fichero y cerrar el fichero. Los VIs y las funciones de bajo nivel pueden gestionar la mayoría de las necesidades de entrada/salida de ficheros. Consultar el LabVIEW Basis II: Development Course Manual para una información más detallada sobre las funciones avanzadas de entrada/salida de ficheros.

VIs Express para File I/O

Los VIs Express en la paleta `File I/O` incluyen los VI Express `Read LabVIEW Measurement File` y el `Write LabVIEW Measurement File`. El fichero de datos de medida de LabVIEW (`.lvm`) es un fichero de texto delimitado por tabs que se puede abrir con una hoja de calculo o un editor de texto. Además de los datos que un Express VI genera, el fichero .lvm incluye información sobre los datos, tales como la fecha y la hora en que fueron generados.

Consulta la sección Data Acquisition and Waveforms, para obtener más información sobre los instrumentos virtuales File I/O Express.

El directorio LabVIEW Data



Usar el directorio por defecto LabVIEW Data para almacenar los ficheros de datos que genera LabViEW, tales como los ficheros `.lvm` y `.txt` LabVIEW crea el directorio `LabVIEW Data` en el mismo directorio en que se encuentra el sistema operativo para facilitar organizar y encontrar los ficheros de datos que LabVIEW genera. El VI Express `Write LabVIEW Measurement File` almacena los ficheros `.lvm` que genera en ese directorio y el instrumento virtual Express `Read LabVIEW Measurement File` lee desde ese directorio. La constante `Default Data Directory`, y la propiedad Default Data Directory también devuelven el directorio `LabVIEW Data` por defecto.

Seleccionar `Tools>>Options` y seleccionar `Paths` en el menú superior que se despliega para especificar un directorio de datos diferente. El directorio de datos por defecto difiere del directorio por defecto, que es el directorio que hay que especificar para los nuevos VIs, controles de usuario, plantillas VI, u otros documentos LabVIEW que vayan a ser creados.

Operaciones básicas de I/O

Una operación típica de entrada/salida implica el siguiente proceso:

1. Crear o abrir un fichero. Indicar donde reside un fichero existente o donde se quiere crear un nuevo fichero, especificando un camino o respondiendo a una caja de diálogo para dirigir a LabVIEW hasta la ubicación del fichero. Después que el fichero se abre, se representa al fichero mediante un numero de referencia, o **refnum**. Este es un identificador único para un objeto, como puede ser un fichero, un dispositivo o una conexión de red.
2. Leer desde o escribir en el fichero.
3. Cerrar el fichero.

VIs de alto nivel para File I/O
En este módulo, se aprenderá a usar los VIs de alto nivel para entrada y salida con ficheros.

La mayoría de los VIs y funciones de File I/O ejecutan solo una fase de la operación de entrada/salida a fichero. Sin embargo, algunos VIs de File I/O de alto nivel diseñados para las operaciones de entrada/salida a ficheros ejecutan todos los pasos. Aunque esos VIs no son siempre tan eficientes como las funciones de bajo nivel, seguramente resultarán más fáciles de usar.

Usar los VIs de File I/O de alto nivel situados en la fila superior de la paleta File I/O para ejecutar tareas habituales de entrada/salida, tales como escribir o leer los siguientes tipos de datos:

- Caracteres a o desde ficheros de texto
- Líneas a o desde ficheros de texto
- Arrays de una o dos dimensiones de valores numéricos de simple precisión a o desde hojas de cálculo
- Arrays de una o dos dimensiones de valores numéricos de simple precisión o enteros con signo de 16 bits a o desde ficheros binarios

Los VIs para entrada/salida de ficheros de alto nivel incluyen lo siguiente:

- **Escritura en una hoja de cálculo** Convierte un array de una o dos dimensiones de números de precisión simple a cadena de texto y escribe la cadena a un nuevo fichero de secuencia de bytes o añade la cadena a un fichero existente. También puede transponer los datos. El VI abre o crea el fichero antes de escribir en él y lo cierra a continuación. Este VI permite crear un fichero de texto que se puede leer con la mayoría de aplicaciones de hoja de cálculo.
- **Lectura desde una hoja de cálculo** Lee un número determinado de líneas o filas desde un fichero comenzando en la posición indicada y convierte los datos a un array de 2 dimensiones de números de simple precisión. El VI abre el fichero antes de leer de él y lo cierra a continuación. Se puede usar este VI para leer un fichero de hoja de calculo salvado en formato texto.

- **Escritura de caracteres a fichero** Escribe una cadena de caracteres a un nuevo fichero de secuencia de bytes o añade la cadena a un fichero existente. El VI abre o crea el fichero antes de escribir en él y lo cierra a continuación.
- **Lectura de caracteres desde fichero** Lee un número determinado de caracteres desde un fichero de secuencia de bytes comenzando a partir de start of read offset. El VI abre el fichero antes de leer de él y lo cierra a continuación.
- **Lectura de líneas desde fichero** Lee un número determinado de líneas desde un fichero binario o de texto comenzando a partir de un carácter determinado. El VI abre el fichero antes de leer de él y lo cierra a continuación.
- **Vis para ficheros binarios** Estos VIs leen desde o escriben en fichero binarios. Los datos pueden ser enteros o números de simple precisión.

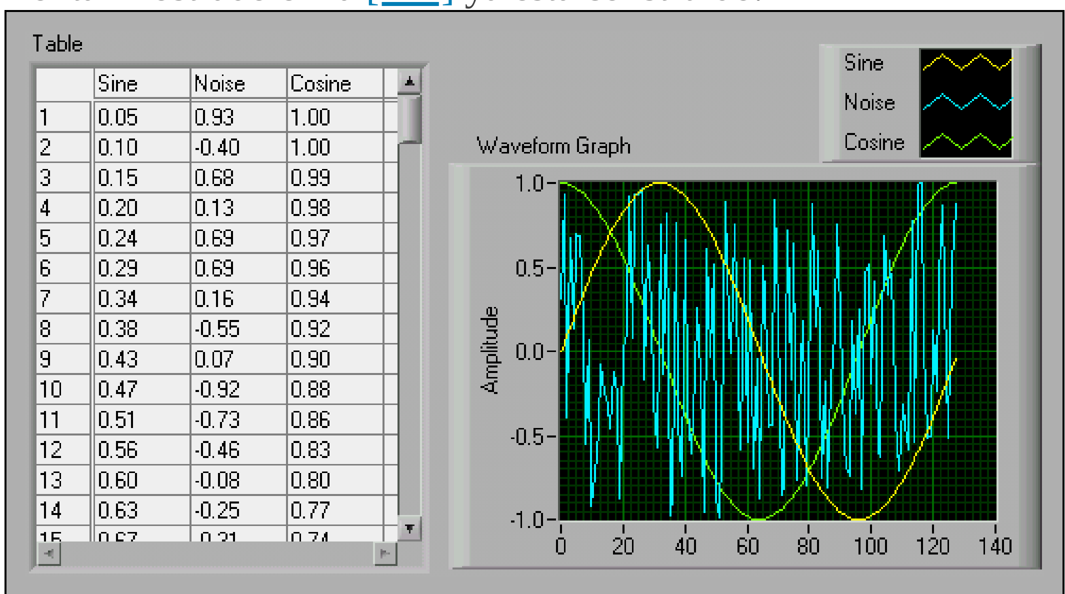
VI ejemplo para trabajar con hojas de cálculo
En este ejercicio, se pretende salvar un array 2D en un fichero de texto para que una hoja de cálculo pueda acceder a los datos y visualizar dichos datos numéricos en una tabla. Completar los siguientes pasos para evaluar el funcionamiento de un VI que guarda arrays numéricos en un fichero en un formato accesible para una hoja de cálculo.

Exercise:

Problem:

Panel Frontal

1. Abrir el VI ejemplo `Spreadsheet Example`, situado en el directorio `C:\Exercises\LabVIEW Basics I`. El panel frontal mostrado en la [link] ya está construido.



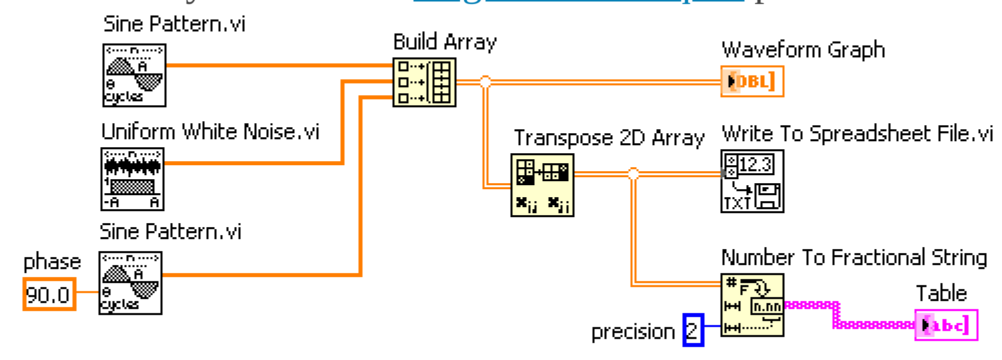
Ejecutar el VI

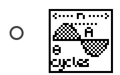
1. Ejecutar el VI. El VI genera un array de 2 dimensiones de $128 \text{ filas} \times 3 \text{ columnas}$. La primera columna contiene los valores correspondientes a una señal senoidal, la segunda columna contiene los datos de una señal de ruido y la 3ª columna contiene datos de una señal cosenoidal. El VI dibuja cada columna en un grafico y muestra los datos en una tabla.

2. Cuando se muestre la caja de dialogo `Choose file to write`, salvar el fichero como `wave.txt` en el directorio `C:\Exercises\LabVIEW Basics I` y hacer clic en el botón `OK`. Después se examinará este fichero.

Diagrama de Bloques

1. Visualizar y examinar el diagrama de bloques para este VI.

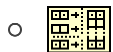


   - 

     El VI `Sine Pattern` situado en la paleta `Functions>>All Functions>>Analyze>>Signal Processing>>Signal Generation` devuelve un array numérico de 128 elementos conteniendo un patrón senoidal. La constante `90.0`, en el segundo VI `Sine Pattern`, especifica la fase de la señal patrón seno o del patrón coseno.

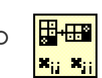
   - 

     El VI `Uniform White Noise` situado en la paleta `Functions>>All Functions>>Analyze>>Signal Processing>>Signal Generation` devuelve un array numérico de 128 elementos conteniendo un patrón de una señal de ruido.

- 

  La función `Build Array` situado en la paleta `Functions>>All Functions>>Array` construye un array de 2 dimensiones para el array seno, el array ruido, y el array coseno, según se muestra en la [link].

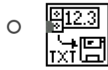
  Sine Array ...
  Noise Array ...
  Cosine Array ...

- 

  La función `Transpose 2D Array` situada en la paleta `Functions>>All Functions>>Array` palette reordena los elementos del array de dos dimensiones tal que $\{i, j\}$ se convierten en $\{j, i\}$, como se muestra en la [link].

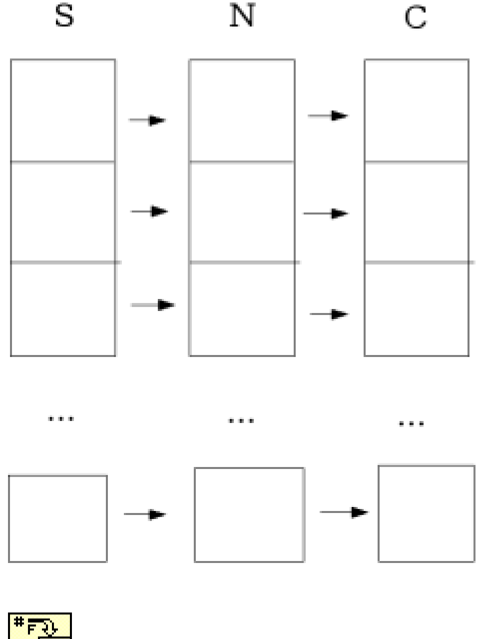
  S N C

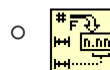
  ... ...

- 

  El VI `Write To Spreadsheet File` situado en la paleta `Functions>>All Functions>>File I/O` da

formato al array 2D en una cadena de hoja de cálculo y escribe la cadena a un fichero. La cadena tiene el formato mostrado en la [link], donde la flecha indica un tabulador y el símbolo de párrafo (¶) indica un carácter de fin de línea.



- 

  La función `Number To Fractional String` situada en la paleta `Functions>>All Functions>>String>>String/Number Conversion` convierte un array de valores numéricos a un array de cadenas de texto tal como se muestra en la tabla.

2. Cerrar el VI. No salvar los cambios.

> **Note:**Este ejemplo solo almacena 3 arrays en el fichero. Para incluir más arrays, incrementar el número de entradas de la función `Build Array`.

Opcional

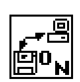
Abrir el fichero `wave.txt` con un editor de texto o una hoja de cálculo y observar su contenido.

1. Ejecutar un editor de texto o una hoja de cálculo, tales como (**Windows**) Notepad, WordPad, UltraEdit, (**Mac OS**) SimpleText, o (**UNIX**) Text Editor.
2. Abrir `wave.txt`. Los datos de la señal senoidal aparecen en la primera columna, los datos de la señal aleatoria en la segunda columna, y los datos correspondientes a la señal coseno en la tercera columna.
3. Salir del editor o de la hoja de cálculo.

VIs y funciones de bajo nivel para File I/O
En este módulo, se aprenderá a usar los VIs y funciones de bajo nivel para la entrada/salida con ficheros.

Usar los siguientes VI y funciones de File I/O de bajo nivel para ejecutar tareas básicas de entrada/salida con ficheros:

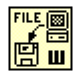
- **Open/Create/Replace File**

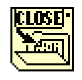
  Abre un fichero existente, crea un fichero nuevo, o sustituye un fichero existente, mediante un programa o interactivamente usando una caja de diálogo de fichero. Opcionalmente se puede especificar un mensaje de diálogo, nombre del fichero por defecto, camino de comienzo, o patrón de filtrado. Si file path está vacío, el VI visualiza una caja de diálogo desde la cual se puede seleccionar un fichero.
- **Read File**

  Lee datos desde el fichero abierto especificado por refnum y los devuelve en data. La lectura comienza en la marca del fichero actual o en una localización especificados por pos mode y pos offset. El modo en que se leen los datos, depende del formato del fichero especificado.
- **Write File**

  Escribe datos a un fichero abierto especificado por refnum. La escritura comienza en la posición especificada por pos mode y pos offset para los ficheros de secuencia de bytes y al final del fichero para los ficheros de datalog. Data, header y el formato del fichero especificado determinan la cantidad de datos escritos.
- **Close File**

Cierra el fichero abierto indicado por refnum y devuelve el camino al fichero asociado con refnum. Error I/O funciona solamente para esta función, la cual cierra el fichero si ha ocurrido un error en una operación precedente. Esto asegura que los ficheros se cierran correctamente. El modo en que se leen los datos, depende del formato del fichero especificado.

Gestión de errores



Los VIs y funciones de bajo nivel con ficheros de entrada/salida devuelven información de error. Conectar la informacion de error desde el comienzo del VI hasta el final. Incluir un VI gestor de errores, tal como el VI Simple Error Handler situado en la paleta `Time & Dialog`, al final del VI para determinar si el VI se ejecutó sin errores. Usar los clusters de entrada y salida de error en cada VI que se use o construya para pasar información del error a través del VI.

Guardar datos en un fichero nuevo o en uno ya existente

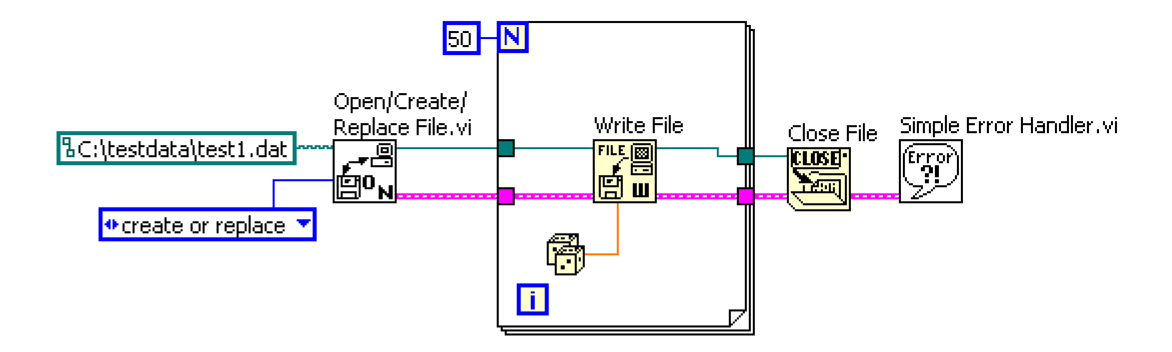
Se puede escribir cualquier tipo de datos a un fichero que se abra o se cree con los VIs y funciones con ficheros de entrada/salida. Si otros usuarios o aplicaciones necesitan acceder al fichero, los datos deberán ser escritos en el fichero en formato ASCII. Consultar el manual *LabVIEW Basics II: Development Course Manual* para obtener mayor información sobre escritura de datalogs de LabVIEW o ficheros binarios.

Se puede acceder a los ficheros mediante programación o interactivamente mediante cajas de diálogo. Para acceder a fichero mediante caja de diálogo, no conectar `file path` en el VI `Open/Create/Replace File`. Sin embargo, se puede ahorrar tiempo conectando el nombre del camino por defecto al VI. La [link] explica como se organizan los nombres de fichero.

Plataforma	Camino
Windows	Está formado por el nombre de la unidad, el carácter dos puntos, los nombres de los directorios separados por backslash y el nombre del fichero. Por ejemplo, `c:\testdata\test1.dat` es el camino para el fichero `test1.dat` en el directorio `testdata` .
UNIX	En UNIX el camino se forma separando los nombres de directorio con el caracter slash y el nombre del fichero. Por ejemplo, `/home/testdata/test1.dat` es el camino para el fichero `test1.dat` en el directorio `testdata` en el directorio `/home` . Los nombres de fichero y los caminos diferencian mayúsculas y minúsculas.

Plataforma	Camino
Mac OS	Se forma con el nombre del volumen (el nombre del disco), el carácter dos puntos, los nombres de las carpetas separados por dos puntos y el nombre del fichero. Por ejemplo `Hard Disk:testdata:test1.dat` es el camino para el fichero llamado `test1.dat` en una carpeta llamada `testdata` en un disco llamado `Hard Disk` .

El diagrama de bloques mostrado en la [link] indica como escribir cadenas de datos a un fichero una vez que se ha especificado el nombre y el camino del fichero. Si el fichero ya existe, será reemplazado; en cualquier otro caso se creará un fichero nuevo.



El VI `Open/Create/Replace File` abre el fichero `test1.dat`. El VI también genera un `refnum` y un cluster de error.

Cuando se abre un fichero, dispositivo o conexión de red, LabVIEW crea un `refnum` asociado con ese fichero, dispositivo o conexión de red. Todas las operaciones que se ejecuten sobre ficheros, dispositivos o conexiones de red abiertos usan `refnum` para identificar cada objeto.

El cluster de error y `refnum` pasa en secuencia de un nodo al siguiente. Dado que un nodo no se puede ejecutar hasta que reciba todas sus entradas, pasando estos dos parametros se fija un orden de ejecución y se crea una dependencia de datos. El VI `Open/Create/Replace File` pasa el refnum y el cluster de error a la función `Write File` que escribe los datos al disco. Cuando la función `Write File` finaliza su ejecución pasa el refnum y el cluster de error a la función `Close File`, que cierra el fichero. El VI `Simple Error Handler` examina el cluster de error y muestra una caja de diálogo si ocurriera un error. Si el error ocurre en un nodo, los nodos subsecuentes no se ejecutan, y el VI pasa el cluster de error al VI `Simple Error Handler`.

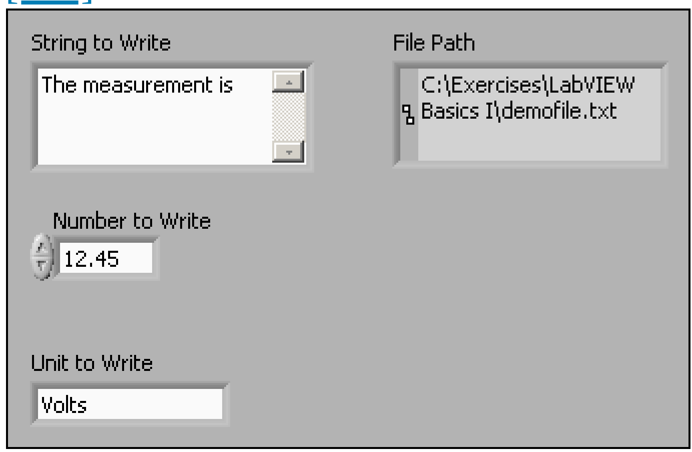
VI para escribir en ficheros
En este ejercicio, se pretente escribir datos a un fichero. Completar los siguientes pasos para construir un VI que concatene un texto, un valor numérico y otro texto y guarde estos datos en un fichero. En otro ejercicio, se construirá un VI que lea el contenido de un fichero y lo muestre al usuario.

Exercise:

Problem:

Panel Frontal

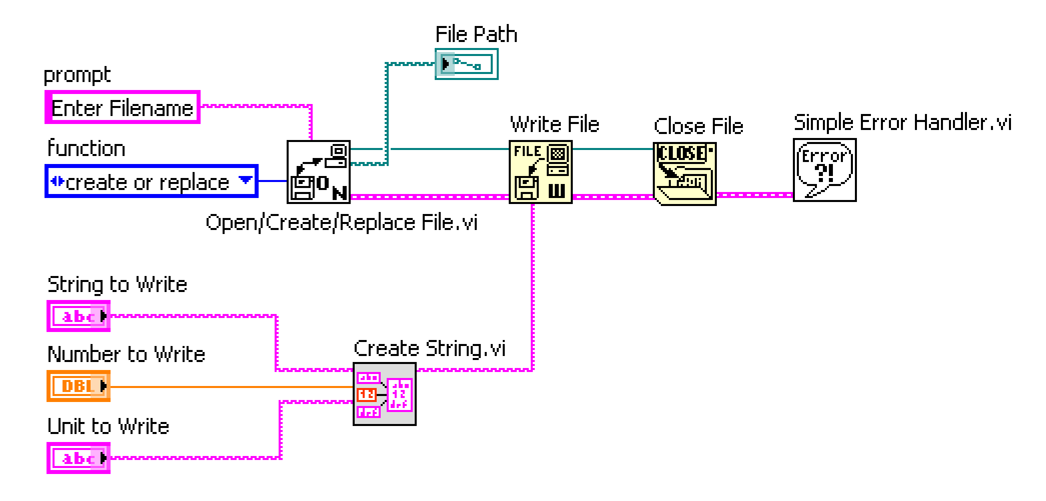
1. Abrir un VI en blanco y construir el panel frontal mostrado en la [link].



   1. Colocar en el panel frontal un indicador de path situado en la paleta `Controls>>Text Indicators`. Este indicador muestra el path para el fichero de datos que se va a crear.
   2. Hacer clic con el botón derecho sobre el control `String to Write` y seleccionar `Visible Items>>Scrollbar` en el menú contextual para mostrar una barra de desplazamiento.

Diagrama de Bloques

1. Construir el diagrama de bloques mostrado en la [link].



1. Colocar en el diagrama de bloques el VI `Create String` del [ejercicio anterior](). Seleccionar `Functions>>All Functions>>Select a VI` y navegar a `C:\Exercises\LabVIEW Basics I\Create String.vi`. Este subVI concatena las tres cadenas de entrada en una cadena combinada.

2. Colocar en el diagrama de bloques el VI `Open/Create/Replace File`, situado en la paleta `Functions>>All Functions>>File I/O`. Este VI muestra una caja de diálogo para abrir o crear un fichero. Hacer clic con el botón derecho del ratón en la entrada de `prompt`, seleccionar `Create>>Constant` en el menú contextual, y escribir `Enter Filename` en la constante. Cuando el VI se ejecuta, aparece una caja de diálogo de navegación de ficheros con `Enter Filename` como título de la ventana. Hacer clic con el botón derecho del ratón en la entrada `function`, seleccionar `Create>>Constant` en el menú contextual, y hacer clic sobre la constante con la herramienta de operaciones para seleccionar `create or replace`.

    3. Colocar en el diagrama de bloques la función `Write File`, situada en la paleta `Functions>>All Functions>>File I/O`. Esta función escribe las cadenas concatenadas en el fichero.
    4. Colocar en el diagrama de bloques la función `Close File`, situada en la paleta `Functions>>All Functions>>File I/O`. Esta función cierra el fichero.
    5. 

       Colocar en el diagrama de bloques el VI `Simple Error Handler`, situado en la paleta `Functions>>All Functions>>Time & Dialog`. Esta función chequea el cluster de error y muestra una caja de diálogo si ocurre un error.
    6. Completar el diagrama de bloques como se muestra en la [link].
2. Salvar el VI como `File Writer.vi` en el directorio `C:\Exercises\LabVIEW Basics I`.

Puesta en marcha del Instrumento Virtual

1. Introducir valores en los controles del panel frontal y ejecutar el VI. Aparece la caja de diálogo `Enter Filename`.
2. Escribir `demofile.txt` y hacer clic sobre el botón `Save` u `OK` para salvar el fichero. El VI escribe los valores `String to Write`, `Numeric to Write`, y `Unit to Write` en el fichero.
3. Cerrar el VI.

VI para leer desde ficheros
En este ejercicio, se pretende construir un VI que lea datos desde un fichero. Completar los siguientes pasos para construir un VI que lea el fichero creado previamente y muestre la información leída en un string indicator.
Exercise:

Problem:

Panel Frontal

1. Abrir un VI en blanco y construir el panel frontal mostrado en la [link] usando el control file path situado en la paleta `Controls>>Text Controls` y un indicador de cadena situado en la paleta `Controls>>Text Indicators`.

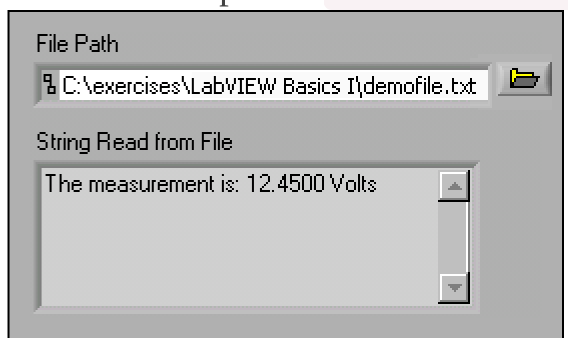
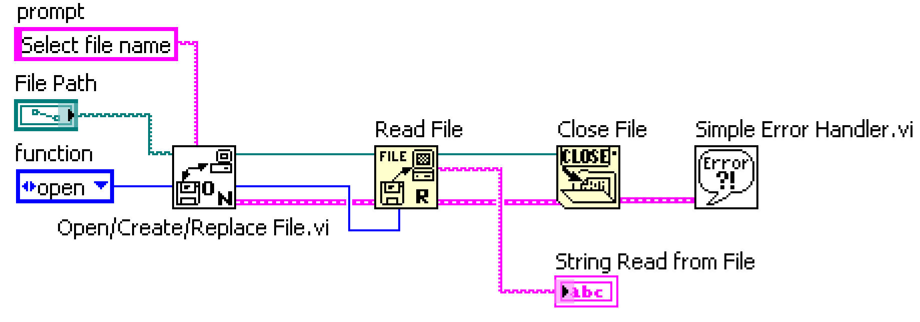


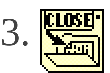
Diagrama de Bloques

1. Construir el diagrama de bloques según se muestra en la [link].



1. 

Colocar en el diagrama de bloques el VI `Open/Create/Replace File`, situado en la paleta `Functions>>All Functions>>File I/O`. Este VI muestra una caja de diálogo para abrir o crear un fichero. Hacer clic con el botón derecho en la entrada `prompt`, seleccionar `Create>>Constant` en el menú contextual, y escribir `Select Filename` en la constante. Hacer clic con el botón derecho en la entrada `function`, seleccionar `Create>>Constant` en el menú contextual, y hacer clic sobre la constante con la herramienta de operaciones y seleccionar `open`.

2. Colocar en el diagrama de bloques la función `Read File`, situada en la paleta `Functions>>All Functions>>File I/O`. Esta función lee un número determinado de bytes (`count`) desde el fichero comenzando por el principio del fichero.
3. 

   Colocar en el diagrama de bloques la función `Close File`, situada en la paleta `Functions>>All Functions>>File I/O`. Esta función cierra el fichero.
4. 

   Colocar en el diagrama de bloques el VI `Simple Error Handler`, situado en la paleta `Functions>>All Functions>>Time & Dialog`. Este VI chequea el cluster de error y muestra una caja de diálogo si ocurre un error.
5. Completar el diagrama de bloques como se muestra en la [link].

2. Salvar el VI como `File Reader.vi` en el directorio `C:\Exercises\LabVIEW Basics I` directory.

Puesta en marcha del Instrumento Virtual

1. Mostrar el panel frontal y usar la herramienta de operaciones para hacer clic sobre el botón `Browse` en el control path.
2. Navegar hasta `demofile.txt` y hacer clic en el botón `Open` or `OK` button.
3. Ejecutar el VI. `String Read from File` muestra el contenido del fichero.
4. Si se dispone de tiempo, se puede completar el siguiente ejercicio challenge step. Si no salvar y cerrar el VI

Desafío

1. Modificar el VI tal que extraiga el valor numérico y lo visualice en un indicador numérico. Una vez finalizado, salvar y cerrar el VI.

> **Note:** Usar la función `Match Pattern` para buscar el primer carácter numérico

Formateado de cadenas de texto para trabajar con Hojas de Cálculo
En este módulo, se aprenderá a dar formato a ficheros de texto para usarlos en hojas de cálculo.

Para escribir datos en un fichero de hoja de cálculo, hay que formatear las cadenas como cadenas de hoja de cálculo, que son cadenas que incluyen delimitadores como por ejemplo tabuladores (tabs). En muchas hojas de cálculo, el carácter tab separa las columnas, y el carácter fin de línea separa las filas.

Note: Usar la constante fin de línea situada en la paleta

```
Functions>>All
      Functions>>String
```

para asegurar la portabilidad de VIs entre diferentes plataformas. (**Windows**) La constante inserta un retorno de carro y un avance de línea. (**Mac OS**) La constante inserta un retorno de carro. (**UNIX**) La constante inserta un avance de línea.

Usar el VI `Write To Spreadsheet File` o la función `Array To Spreadsheet String` para convertir un conjunto de números desde un gráfico, un diagrama o una adquisición en una cadena de hoja de cálculo. Si se quiere escribir números y textos para una hoja de cálculo o un procesador de textos, usar las funciones de String y las funciones de Array para formatear los datos y combinar las cadenas. Después escribir los datos en un fichero.

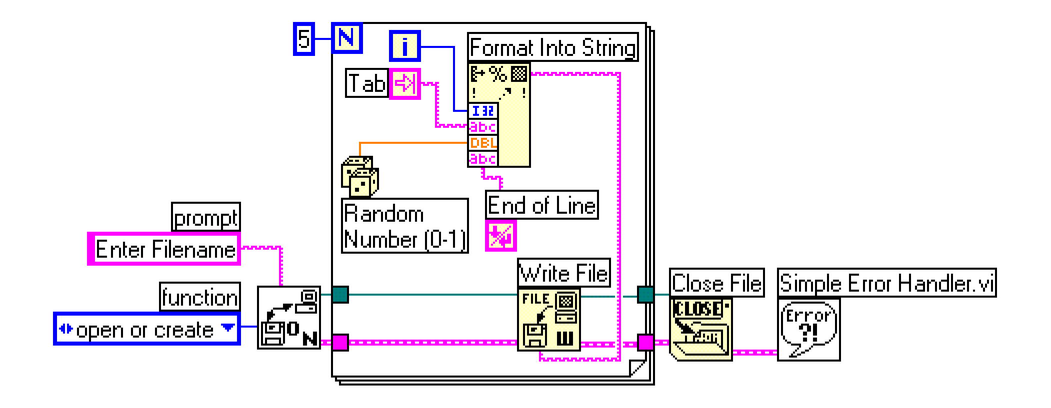
Format Into File

Usar la función `Format Into File` para dar formato a cadenas, numeros, caminos de acceso y datos Boleanos como texto y escribir el texto a un fichero. A menudo se puede usar esta función en vez de hacerlo por

separado, dar formato a la cadena con la función `Format Into String` o con el VI Express `Build Text` y escribir la cadena resultante con el VI `Write Characters To File` o con la función `Write File`.

Usar la función `Format Into File` para establecer el orden en que aparecen los datos en el fichero de texto. Sin embargo, no se puede usar esta función para añadir datos a un fichero o sobreescribir los datos existentes en un fichero. Para esas operaciones usar la función `Format Into String` junto con la función `Write File`. Se puede conectar un `refnum` o `path` al terminal `input file` de la función `Format Into File`, o se puede dejar esta entrada sin conectar y mediante una caja de diálogo, se preguntará al usuario el nombre del fichero.

En el diagrama de bloques mostrado en la [link], el VI `Open/Create/Replace File` abre un fichero, y el `bucle For` se ejecuta 5 veces. La función `Format Into File` convierte el número de repeticiones y el número aleatorio en cadenas de texto y coloca los caracteres tabulador y fin de línea en las posiciones correctas para crear 2 columnas y una fila en formato de hoja de calculo. Después que el bucle completa las 5 repeticiones, el fichero se cierra, y el VI chequea la condición de error.



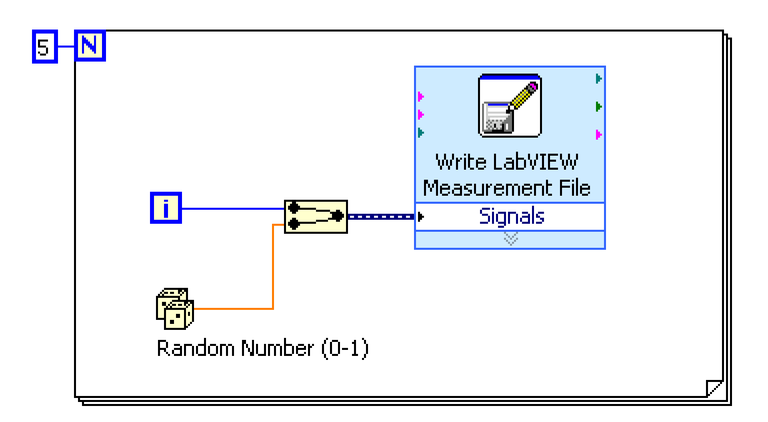
Este VI crea este fichero de texto, en el que una flecha (→) indica un tabulador, y el símbolo de fin de párrafo (¶) indica un carácter de final de línea. `0→ 0.798141¶ 1→ 0.659364¶ 2→ 0.581409¶ 3→ 0.526433¶ 4→ 0.171062¶`

Se puede abrir el fichero de texto anterior en una aplicación de hoja de cálculo para visualizarla como se muestra a continuación en la [link].

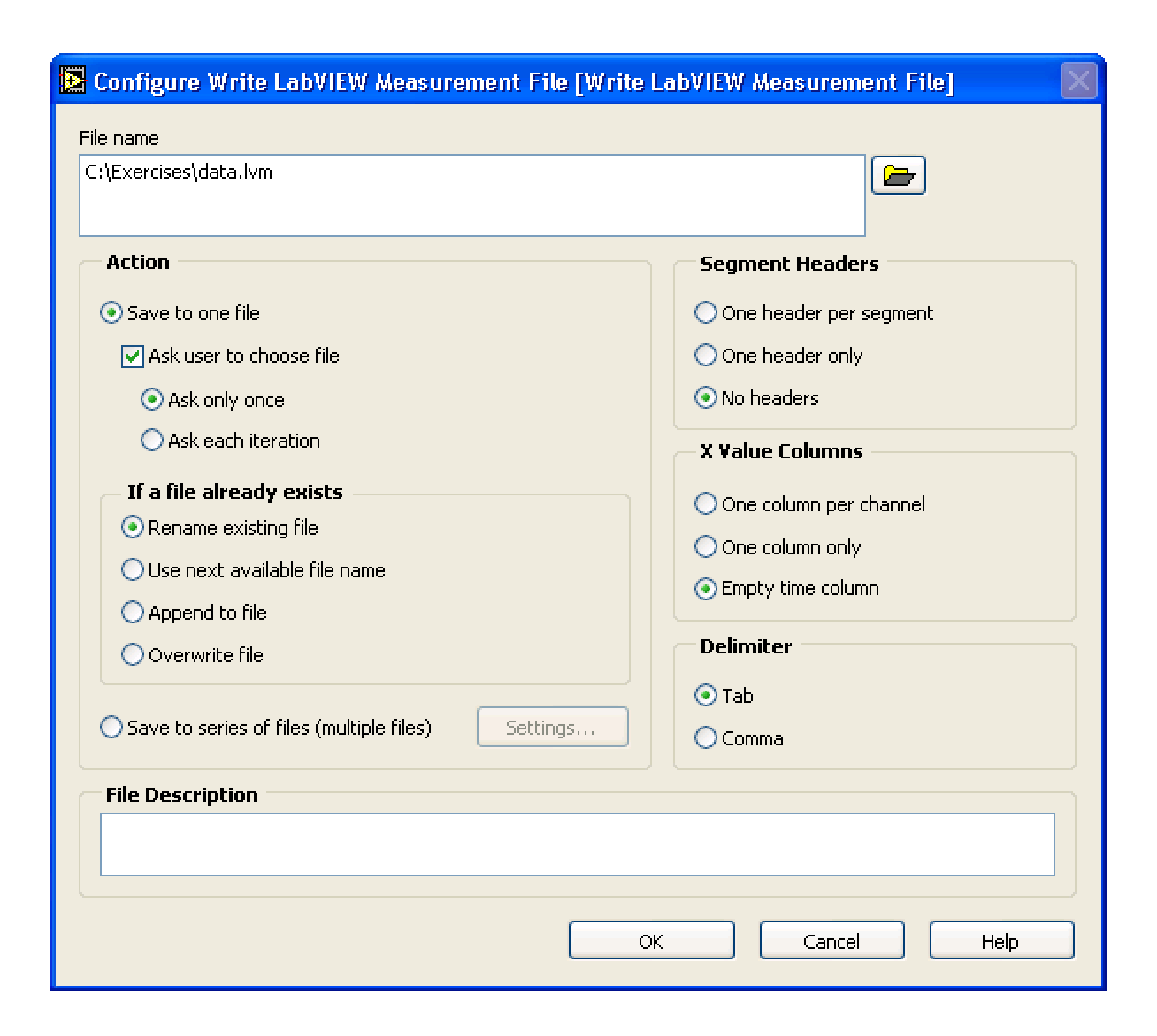
	A	B
1	0	0.798141
2	1	0.659364
3	2	0.581409
4	3	0.526433
5	4	0.171062

Escribir un fichero de datos de LabVIEW

En el diagrama de bloques mostrado en la [link], el VI Express `Write LabVIEW Measurement File` incluye las funciones abrir, escribir, cerrar y gestión de errores. También se encarga del formateado de la cadena de texto con tabuladores o delimitado por comas. La función `Merge Signals` combina el contaje de repetición del bucle y el número aleatorio en un tipo de datos dinámico.



La caja de diálogo en la [link] muestra la configuración para el VI Express `Write LabVIEW Measurement File`.



Este VI crea un fichero `.lvm` que se puede abrir con una hoja de cálculo. La [link] muestra un ejemplo de la hoja de cálculo creada por el VI Express `Write LabVIEW Measurement File`, haciendo uso de la configuración mostrada anteriormente.

	A	B	C	D
1		0	0.385055	
2		1	0.23516	
3		2	0.985184	
4		3	0.177893	
5		4	0.935915	
6				
7				

Consultar el módulo Adquisición de datos y formas de onda (Data Acquisition and Waveforms), para obtener más información de los instrumentos virtuales Express `Write LabVIEW Measurement File` y `Read LabVIEW Measurement File`.

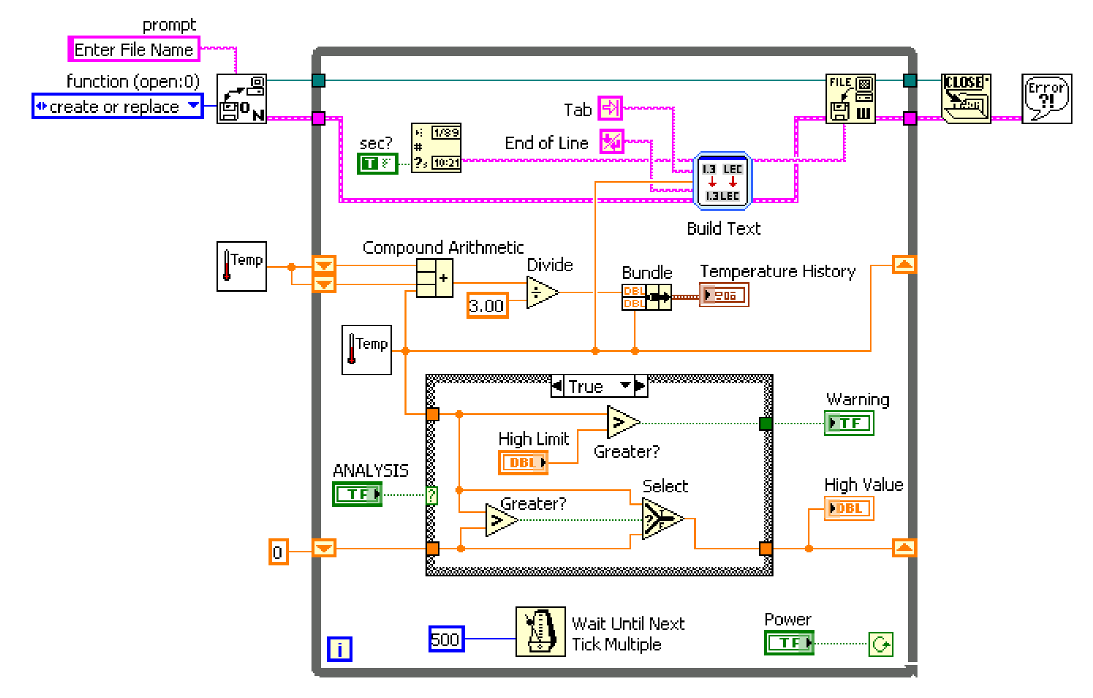
VI Registrador de temperatura
En este ejercicio, se pretende salvar datos a un fichero de forma se pueda acceder desde una hoja de cálculo o desde un procesador de texto. Completar los pasos siguientes para construir un VI que guarde la hora y la temperatura en un fichero de datos.

Exercise:

Problem:

Block Diagram

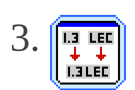
1. Abrir el Vi Temperature Control VI y salvarlo como `Temperature Logger.vi` en el directorio `C:\Exercises\LabVIEW Basics I`. No hace falta modificar el panel frontal.
2. Abrir y modificar el diagrama de bloques según se muestra en la [link]. Cambiar el tamaño del `bucle While` para añadir espacio en la parte superior para las operaciones de entrada/salida con ficheros.



1.

Colocar en el diagrama de bloques el VI `Open/Create/Replace File`, situado en la paleta `Functions>>All Functions>>File I/O`. Hacer clic con el botón derecho sobre la entrada `prompt`, seleccionar en el menú contextual la opción `Create Constant` y escribir `Enter File Name` en la constante. Hacer clic con el botón derecho sobre la entrada `function`, seleccionar `Create Constant` en el menú contextual, y hacer clic sobre la constante con la herramienta de operaciones para seleccionar `create or replace`.

2. Colocar en el diagrama de bloques la función `Get Date/Time String`, situada en la paleta `Functions>>All Functions>>Time & Dialog`. Esta función devuelve la hora en formato cadena de texto cada vez que se mide la temperatura. Hacer clic con el botón derecho en la entrada `want seconds?`, seleccionar en el menú contextual `Create>>Constant`, y hacer clic sobre dicha constante para cambiar el valor de `Falso` a `Verdadero`. Un valor de verdadero en la entrada `want seconds?` hace que la función incluya los segundos en la cadena de texto.

3. 

   Colocar en el diagrama de bloques el VI Express `Build Text`, situado en la paleta `Functions>>Output`. Este VI Express convierte las entradas a una cadena de texto. Aparece la caja de diálogo `Configure Build Text`.

   1. En la caja de texto `Text with Variables in Percents` escribe la cadena `%tab%%temp%%end%` para fijar las tres variables; una para la constante tabulador, otra para la temperatura y otra para la

   constante fin de línea. Dado que el tiempo usa la entrada `Beginning Text` del VI Express `Build Text`, no se necesita ninguna variable.
2. Seleccionar la temperatura en la sección `Configure Variables`. Seleccionar la opción `Number`, y un formato de `Format fractional number`. Las variables `tab` y `end` no necesitan ser formateadas. Se pueden dejar con su valor por defecto.
3. Hacer clic sobre el botón `OK` para cerrar la caja de diálogo de configuración.
4. Hacer clic con el botón derecho sobre el VI Express `Build Text` y seleccionar `View As Icon` para ahorrar espacio en el diagrama de bloques.

4.



Colocar en el diagrama de bloques una `constate tabulador` y una `constante fin de línea`, situadas en la paleta `Functions>>All Functions>>String`.

5.

Colocar en el diagrama de bloques la función `Write File`, situada en la paleta `Functions>>All Functions>>File I/O`. Esta función escribe en el fichero especificado por `refnum`.

6.

Colocar en el diagrama de bloques la función `Close File`, situada en la paleta `Functions>>All Functions>>File I/O`. Esta función cierra el fichero.

7. 

Colocar en el diagrama de bloques el instrumento virtual `Simple Error Handler`, situado en la paleta `Functions>>All Functions>>Time & Dialog`. Este VI chequea el cluster de error y visualiza un mensaje de error si ocurriera un error.

8. Completar el diagrama de bloques como se muestra en la [link].

3. Salvar el VI. Este VI se usará más tarde en este curso.

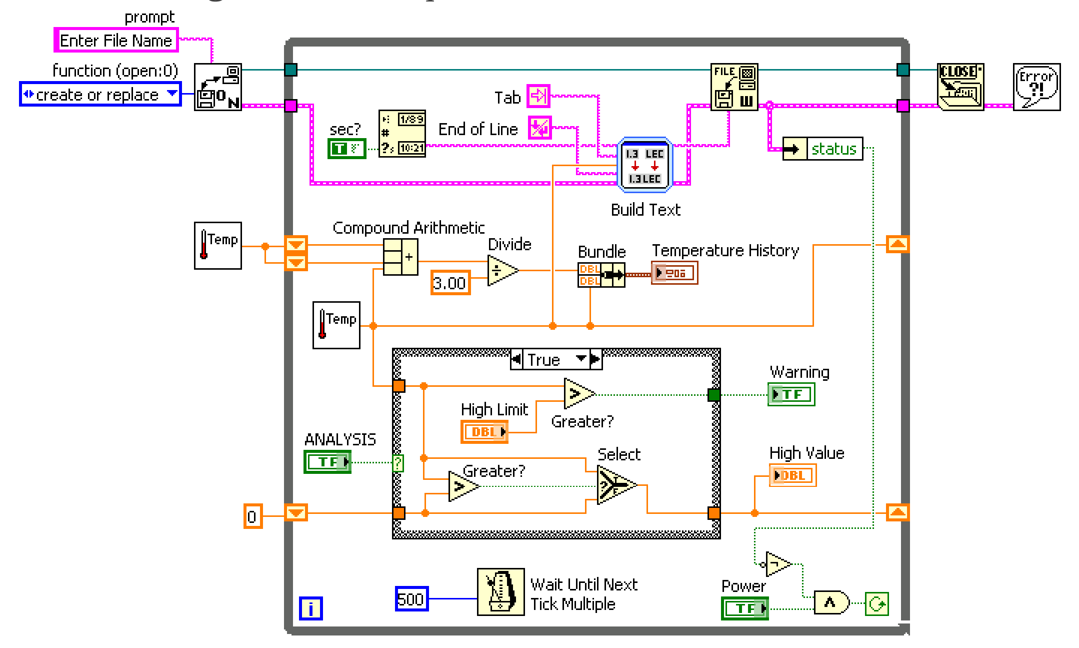
Puesta en marcha del Instrumento Virtual

1. Mostrar el panel frontal y ejecutar el VI. Se muestra la caja de diálogo `Enter File Name`.
2. Escribir `temp.txt` y hacer clic en el botón `Save` u `OK`. El VI crea un fichero llamado `temp.txt`. El VI va tomando lecturas cada medio segundo y salva los datos de hora y temperatura en el fichero hasta que se haga clic en el interruptor `Power`. Cuando el VI termina la ejecución, se cierra el fichero.
3. Abrir un procesador de textos o una aplicación para hojas de calculo, tales como (**Windows**) Notepad, WordPad o UltraEdit, (**Mac OS**) SimpleText, o (**UNIX**) Text Editor.
4. Abrir el fichero `temp.txt` con el procesador de texto o con la hoja de calculo. La hora aparece en la primera columna, y los datos de la temperatura aparecen en la segunda columna.
5. Salir del procesador de texto o la hoja de cálculo y volver a LabVIEW.
6. Si se dispone de tiempo, completar los siguientes pasos. En caso contrario cerrar el VI.

Optional

Cuando se haga uso de gestión de errores en un VI, el `bucle While` debería parar ls ejecución cuando ocurre un error. Completa los pasos siguientes para modificar el VI de modo que se pare cuando el usuario hace clic sobre el interruptor de marcha u ocurra un error.

1. Editar el diagrama de bloques como se muestra en la [link].



   1. status

      Colocar en el diagrama de bloques la función `Unbundle by Name`, situada en la paleta `Functions>>All Functions>>Cluster`. Esta función lee la salida status del cluster de error.

   2. 

      Colocar en el diagrama de bloques las funciones `Not` y `And`, situadas en la paleta `Functions>>Arithmetic & Comparison>>Express Boolean`. Estas funciones hacen que el `bucle While` continue ejecutándose mientras `Power` sea verdadero y no haya errores.

2. Salvar y ejecutar el VI.
3. Testear el gestor de errores borrando la conexión `refnum` entre la función `Write File` y el lateral izquierdo del `bucle While`. Hacer clic con el botón derecho sobre la entrada `refnum` de la función `Write File` y elegir `Create>>Constant`.
4. Ejecutar de nuevo el VI. El VI debería esperar la introducción de un camino, y a continuación parar inmediatamente con un error. Si la gestión de errores no hubiera sido incluida en este VI, el VI no informaría del error hasta que el usuario detuviera la ejecución.
5. Si se dispone de tiempo, completar los siguientes pasos. De otro modo, cerrar el VI. No salvar los cambios.

Desafío

1. Sustituir el VI Express `Build Text` y la función `Write File` con la función `Format Into File`.
2. Ejecutar el VI.
3. Cerrar el VI. No salvar los cambios.

VI Registro de temperatura y representación de los datos
En este ejercicio, se aplicarán los conocimientos adquiridos sobre estructuras, registros de desplazamiento, diagramas para formas de onda, arrays, gráficos, entrada y salida con ficheros, etc.

Exercise:

Problem:

1. Construir un Instrumento Virtual para ejecutar las siguientes tareas:
   1. Tomar una muestra de temperatura cada segundo hasta que se detenga la ejecución o se produzca un error.
   2. Visualizar la temperatura actual y la promediada de las 3 últimas medidas en un waveform chart.
   3. Si la temperatura excede un límite determinado, encender un diodo LED.
   4. Después de cada medida, registrar la fecha, la hora, incluyendo los segundos, la temperatura, la media de las 3 últimas medidas de temperatura y un mensaje de una palabra, describiendo si la temperatura es normal o supera el límite. Registrar los datos, de tal manera que cada elemento aparezca en una columna de la hoja de cálculo como se muestra en la [link].

	A	B	C	D	E
1	Date	Time	Temp	Avg	Comment
2	9/26/00	12:45:17 AM	74.46	74.46	Normal
3	9/26/00	12:45:18 AM	74.46	74.46	Normal
4	9/26/00	12:45:19 AM	74.46	74.46	Normal
5	9/26/00	12:45:20 AM	74.46	74.46	Normal

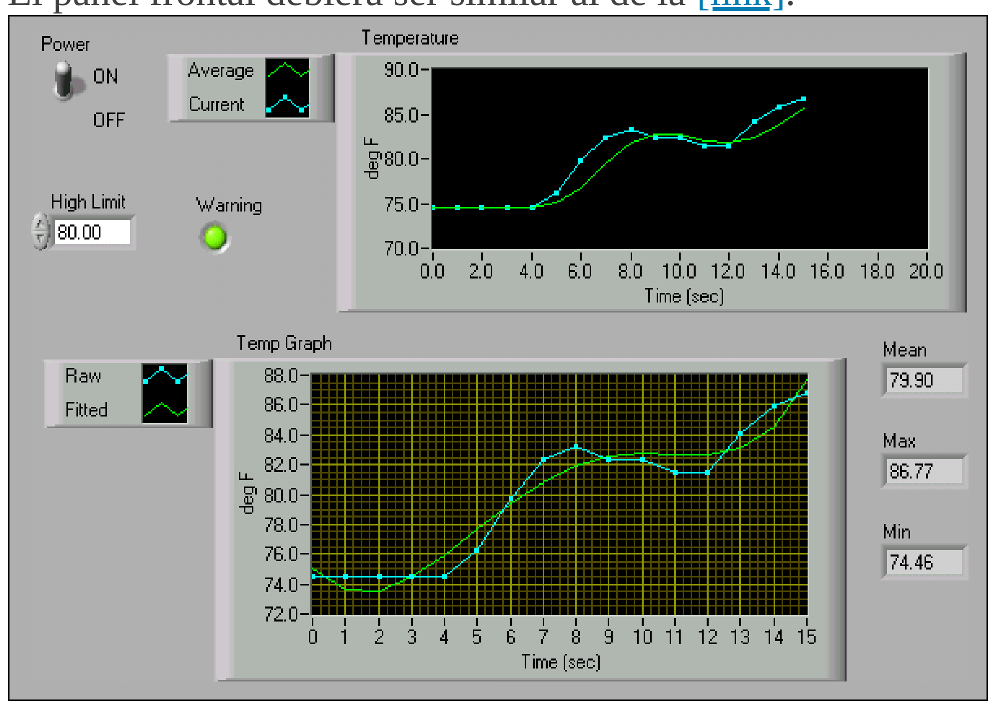
   5. Una vez finalizada la toma de muestras, plotear los valores de temperatura leídos y la curva que mejor se ajuste en un gráfico XY y mostrar las temperaturas media, máxima y mínima.

Note: Comenzar con el VI Temperature Logger. Para completar el ejercicio usar partes del Instrumento Virtual

Análisis de temperatura.

Note: El VI Express `Build Text` está limitado a 8 entradas. Usar varios VIs Express `Build Text` o usar la función `Format into String`. También se podría construir los delimitadores en la cadena de formato para necesitar menos entradas.


El panel frontal debiera ser similar al de la [link].



2. Salvar el VI como `Temperature Application.vi` en el directorio `C:\Exercises\LabVIEW Basics I`.

Resumen, pistas y trucos sobre cadenas de texto y ficheros de Entrada/Salida

- La cadenas de texto agrupan secuencias de caracteres ASCII. Usar el control y el indicador string para simular cajas de entrada de texto y etiquetas.
- Para minimizar el espacio que ocupa un objeto cadena, hacer clic con el botón derecho sobre el objeto y seleccionar `Show Scrollbar` en el menú que aparece.
- Usar las funciones de String situadas en la paleta `Functions>>All Functions>>String` para editar y manipular cadenas de texto en el diagrama de bloques.
- Usar el VI Express `Build Text` para convertir un valor numérico en una cadena de texto.
- Usar la función `Scan From String` para convertir una cadena en un valor numérico.
- Hacer clic con el botón derecho sobre la función `Scan From String` y seleccionar `Edit Scan String` en el menú que aparece para editar o crear una cadena de formato.
- Usar los VIs y funciones para ficheros de Entrada/Salida para manejar todos los aspectos relativos a file I/O.
- Usar los VIs de I/O de alto nivel para ejecutar operaciones comunes de entrada/salida.
- Usar los VIs y las funciones de I/O de bajo nivel y las funciones Advanced File I/O situadas para controlar cada operación de entrada/salida de manera individual.
- Usar los VIs Express de File I/O para operaciones sencillas de captura de datos.
- Cuando se escribe en un fichero, éste se abre, crea, o sobreescribe, se escriben los datos, y se cierra el fichero. De igual manera, cuando se lee desde un fichero, se abre un fichero existente, se leen los datos y se cierra el fichero.
- Para acceder a un fichero mediante una caja de diálogo, dejar desconectado el file path en el instrumento virtual `Open/Create/Replace File`.
- Para escribir datos en un fichero de hoja de calculo, la cadena tiene que estar formateada como una cadena de hoja de calculo, es decir, una

cadena que incluye delimitadores como pueden ser, por ejemplo, los tabuladores. Usar la función `Format Into File` para dar formato a datos de cadena, numéricos, caminos de almacenamiento de ficheros y variables Booleanas y escribir el texto en un fichero.

Ejercicios adicionales para cadenas de texto y Ficheros de entrada y salida

Exercise:

Problem:

Construir un instrumento virtual que genere un array de 2 dimensiones con de números aleatorios y escriba los datos transpuestos en una hoja de calculo. Añade un encabezamiento en cada columna. Usar los VIs de alto nivel de File I/O situados en la paleta File I/O.

Note: Usar el VI

`Write Characters To File`

para escribir el encabezado y el Write To Spreadsheet File VI para escribir los datos numéricos en el mismo fichero.

Salvar el instrumento virtual como `More Spreadsheets.vi` en el directorio `C:\Exercises\LabVIEW Basics I`.

Exercise:

Problem:

Construir un VI que convierta cadenas de hoja de calculo delimitadas por tabuladores a cadenas de hoja de calculo delimitadas por comas, es decir, cadenas de hoja de calculo con columnas separadas por comas y filas separadas por caracteres end of line. Visualiza en el panel frontal ambas cadenas, la delimitada por tabuladores y la delimitada por comas.

Note: Usar la función

```
Search and Replace String
```

.

Salvar el instrumento virtual como `Spreadsheet Converter.vi` en el directorio `C:\Exercises\LabVIEW Basics I`.

Exercise:

Problem:

Modificar el instrumento virtual Temperature Logger VI para que la aplicación no cree un nuevo fichero cada vez que se ejecuta. Añadir los datos al final del fichero existente `temp.dat` que fue creado por el VI `Temperature Logger`. Ejecutar el instrumento virtual varias veces y usar un de procesador de texto para comprobar que el instrumento virtual añade nuevas lecturas de temperatura.

Note: Borrar la función

```
Format Into File
```

y sustituirla con las funciones

```
Format Into String
```

y

```
Write File
```

. Usar los parámetros

```
pos
          mode
```

y

`pos offset`

de la función

`Write File`

para mover la marca de fichero actual.

Seleccionar `File>>Save As` para salvar el instrumento virtual como `Temperature Logger 2.vi` en el directorio `C:\Exercises\LabVIEW Basics I`.